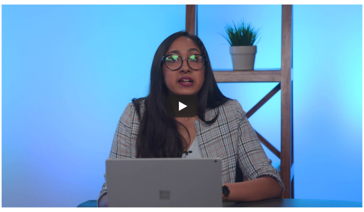
Encoding Categorical Data

As we've mentioned a few times now, machine learning algorithms need to have data in numerical form. Thus, when we have *categorical* data, we need to encode it in some way so that it is represented numerically.

There are two common approaches for encoding categorical data: **ordinal encoding** and **one hot encoding**.



Ordinal Encoding

In **ordinal encoding**, we simply convert the categorical data into integer codes ranging from `0` to `(number of categories - 1)`. Let's look again at our example table of clothing products:

SKU	Make	Color	Quantity	Price
908721	Guess	Blue	789	45.33
456552	Tillys	Red	244	22.91
789921	A&F	Green	387	25.92
872266	Guess	Blue	154	17.56

If we apply ordinal encoding to the `Make` property, we get the following:

Make	Encoding
A&F	0
Guess	1
Tillys	2

And if we apply it to the Color property, we get:

Color	Encoding
Red	0
Green	1
Blue	2

Using the above encoding, the transformed table is shown below:

SKU	Make	Color	Quantity	Price
908721	1	2	789	45.33
456552	2	0	244	22.91
789921	0	1	387	25.92
872266	1	2	154	17.56

One of the potential drawbacks to this approach is that it implicitly assumes an order across the categories. In the above example, `Blue` (which is encoded with a value of `2`) seems to be *more* than `Red` (which is encoded with a value of `1`), even though this is in fact not a meaningful way of comparing those values. This is not *necessarily* a problem, but it is a reason to be cautious in terms of how the encoded data is used.

One-Hot Encoding

One-hot encoding is a very different approach. In one-hot encoding, we transform each categorical value into a column. If there are `n` categorical values, `n` new columns are added. For example, the `Color` property has three categorical values: `Red`, `Green`, and `Blue`, so three new columns `Red`, `Green`, and `Blue` are added.

If an item belongs to a category, the column representing that category gets the value `1`, and all other columns get the value `0`. For example, item 908721 (first row in the table) has the color blue, so we put `1` into that `Blue` column for 908721 and `0` into the `Red` and `Green` columns. Item 456552 (second row in the table) has color red, so we put `1` into that `Red` column for 456552 and `0` into the `Green` and `Blue` columns.

If we do the same thing for the `Make` property, our table can be transformed as follows:

SKU	A&F	Guess	Tillys	Red	Green	Blue	Quantity	Price
908721	0	1	0	0	0	1	789	45.33
456552	0	0	1	1	0	0	244	22.91
789921	1	0	0	0	1	0	387	25.92
872266	0	1	0	0	0	1	154	17.56

One drawback of one-hot encoding is that it can potentially generate a very large number of columns.

QUESTION 1 OF 4

Have a look at this tabular data:

ID	Mammal	Reptile	Fish
012	1	0	0
204	0	0	1
009	0	1	0
105	1	0	0

What type of encoding has been performed on this?

○ Ordinal encoding

✓ One-hot encoding

SUBMIT

QUESTION 2 OF 4

Looking again at the table in the previous question, what category is animal `204`?

○ Mammal

○ Reptile

✓ Fish

SUBMIT

QUESTION 3 OF 4

Again looking at the above animals table, suppose we do the following:

1. Add two new categories, `Amphibian` and `Bird`
2. Add one bird with ID `303` in the table

Which one of the following statements is correct about the new table?

○ There are 5 columns in the new table including the `ID` column

○ Animal `303` has `1` in the `Mammal` column

✓ The `Amphibian` column has `0` for all animals

○ Animal `303` has `0` in the `Bird` column

SUBMIT

QUESTION 4 OF 4

John is looking to train his first machine learning model. One of his inputs includes the size of the T-Shirts, with possible values of XS, S, M, L, and XL. What is the best approach John can employ to preprocess the T-Shirt size input feature?

○ Standardization

○ Normalization

✓ One Hot Encoding

SUBMIT